REMARKS

This paper is submitted in reply to the Office Action dated May 23, 2005. A request for a one month extension of time has been submitted concurrently herewith. Therefore, the period of response extends up to and includes September 23, 2005. Reconsideration and allowance of all pending claims by the Examiner are respectfully requested.

In the subject Office Action, claims 1-32[1] were rejected under 35 U.S.C. §112 second paragraph. Moreover, claims 1-14, 16-29 and 31-32 were rejected under 35 U.S.C. §102(e) as being anticipated by U.S. Patent No. 6,195,676 to Spix et al. (Spix et al.). In addition, claims 15 and 30 were rejected under 35 U.S.C. §103(a) as being unpatentable over Spix et al. in view of U.S. Patent No. 5,978,830 to Nakaya et al. (Nakaya et al.).

Applicants respectfully traverse the Examiner's rejections to the extent that they may be maintained. Claims 28-30 are cancelled to further the remaining claims onto issuance.

As an initial matter, Applicants appreciate the courtesy provided by the Examiner to speak briefly with Applicants' representative on August 31, 2005, during which the differences between the claimed multithreaded CPU and a non-multithreaded CPU in a multithreaded operating environment were discussed. At the conclusion of the discussion, the Examiner appeared to appreciate that these differences, in particular regarding the specific characteristics of a multithreaded CPU, and the issues these characteristics raise in connection with handling yields.

Turning first to the substantive rejections of the Office Action, and more particularly, to the § 102(e) rejection of independent claim 1, this claim generally recites a method for sharing resources on a multithreaded CPU. As described on page 1-2 of the

---

[1] Claim 33, which depends from claim 16, was not addressed in the Office Action.

application, a multithreaded CPU is a processor chip that can execute two or more threads in hardware. The claimed invention specifically addresses programming constraints that have conventionally hindered multithreaded CPU resource sharing. For instance, all threads executing on a multithreaded CPU are required to execute within a common virtual space, such as a partition. The claimed invention enables resource sharing despite such constraints by deferring a yield of a first thread executing on the multithreaded CPU, while waiting for at least a second thread executing on the multithreaded CPU to become ready to yield, and yielding the first thread in response to at least the second thread becoming ready to yield.

Spix et al. fails to teach this claimed feature. In fact, Spix et al. is concerned only with non-multithreaded CPU's. There is no requirement for threads in Spix et al. to execute within a common virtual space, among other programming requirements of the claimed multithreaded CPU, and there is logically no suggestion or teaching of a multithreaded CPU anywhere in cited references.

Moreover, there is no teaching of a yield command in Spix et al. A yield has a definite meaning in the art, but a brief discussion of a yield is included to resolve any lack of clarity as perceived by the Examiner, and to highlight the distinctions between the present claims and the prior art.

In a conventional, non-multithreaded CPU partitioned environments, partitions share computing resources, such as CPU's. As explained in greater detail on pages 2-4 of the application as filed, CPU's within a conventional, non-multithreaded CPU environment are dynamically assigned to handle independent units of execution, in different partitions in the environment. These units of execution are often referred to as virtual processors, and in such environments, yield commands may be made by an operating system when a virtual processor of a partition cannot use its allocated CPU efficiently. For instance, a virtual processor may be in an idle loop or may have no work to do. The yield command causes the virtual processor to surrender its CPU availability

to another virtual processor, and subsequently enter an idle state. Thus, the (non-multithreaded) CPU is used more efficiently by virtue of its allocation to a virtual processor that can utilize it.

While yield commands conventionally work well within multithreaded environments, e.g., where partitions share multiple non-multithreaded CPU's yields have never before the present invention been used with multithreaded CPU's. One reason for this is because multithreaded CPU's have a programmatic limitation that requires all threads to execute within a common virtual space, e.g., a hypervisor or partition. The claimed subject matter enables a thread executing on a multithreaded CPU to yield, and does so in a manner to ensure that all threads in the multithreaded CPU execute within the same virtual space. Specifically, a single thread's yield is deferred until another thread is in a ready-to-yield state, which ensures that the threads can all yield in a common virtual space.

While Spix et al. does disclose one manner of managing resources, it still fails to disclose or suggest a yield command. Significantly, Spix et al. does not once even mention the concept of a "yield" or its equivalent. If the Examiner is aware of anywhere in Spix et al. a yield is disclosed, he is respectfully asked to point particularly to this text.

The (non-yield) methods disclosed in Spix et al. is described at column 7, lines 36-45. The system of Spix et al. attempts to schedule resources "by allowing each processor to access a single image of the operating system" to provide a "visual representation" for a plurality of compiling tools. To this end, the Spix et al. system utilizes microprocessors (mprocs) in addition to a User-Side Scheduler (USS). In the text cited by the Examiner in column 8, the USS is described as prioritizing processes by allowing a mproc to grab a single image of the memory. The USS functions to place work and look for work in a request queue (column 8, lines 55-60). The USS further utilizes a scheme for marking data and/or resources (mark or gmark) (column 45, lines 37-48). In any case, the USS docs not implement or suggest a yield command.

Because Spix et al. discloses neither a multithreaded CPU nor a yield command, Spix et al. consequently fails to teach each and every element of claim 1. Applicants respectfully request that the §102(e) rejection of claim 1 be withdrawn.

Moreover, even if hypothetically combined with a reference having a multithreaded CPU, it would still not be obvious to modify Spix et al. to use multithreaded CPU's because of peculiarities with multithreaded CPU's that are not accommodated in the Spix et al. system. Spix et al. has no appreciation for the need to check the state of one thread in connection with allocating another thread to handle a particular task. As such, Spix et al. would not motivate one of ordinary skill in the art to defer a yield of one thread on a multithreaded CPU based upon the state of another thread. Accordingly, claim 1 is also non-obvious over Spix et al. Reconsideration and allowance of claim 1, as well as of claims 2-12 which depend therefrom, are respectfully requested.

Independent claim 13 recites deferring a yield of a thread within a mulithreaded CPU system while at least a subset of the plurality of threads yield, and abandoning the yield of the thread in response to detecting an event while the yield is deferred. As discussed above, Spix et al. fails to disclose or suggest a yield call, let alone deferring a yield. As such, Applicants respectfully request that the §102(e) rejection of claim 13 be withdrawn. Reconsideration and allowance of claim 1, as well as of claims 14 and 15 which depend therefrom, are respectfully requested.

Independent claim 16 is a computer hardware and software implementation that includes features similar to those recited in claim 1. These features include the multithreaded CPU that is neither taught nor suggested by Spix et al. Applicants consequently request reconsideration and allowance of claim 16, as well as of claims 17-22, 24-25 and 33 that depend therefrom.

Independent claim 31 is a program product implementation that includes features similar to those recited in claim 1. These features similarly include, among others,

deferring a yield of a first thread executing on the multithreaded CPU. Since there is no such teaching present in Spix et al., Applicants respectfully request reconsideration and allowance of claim 31, as well as of claim 32 that depends therefrom.

Regarding the remaining nonsubstantive rejections, Applicants respectfully request that the §112 rejections be withdrawn, as the claims do particularly point out and distinctly claim the subject matter as supported and described in the specification. The above description explaining the purpose and function of a conventional yield call should have resolved any lack of clarity as perceived by the Examiner, in addition to distinguishing the present claims and the cited art. However, for the sake of completeness, Applicants maintain that the claims do particularly point out and distinctly claim the subject matter as supported and described in the specification. A method claim, such as claims 1 and 13, does not have to recite associated structure and supporting steps, so long as the associated structure and processes are described in a specification in a manner that would enable one of skill in the art. *The Regents of the University of California v. Eli Lily & Co.*, 33 F.3d 1526 (Fed. Cir. 1994). The recited language of claims 1 and 13, namely, "deferring a yield," "become ready to yield," "second threads becoming ready to yield," and "subset" is described in an enabling manner throughout the specification (for instance, on pg. 14 at line 7, pg. 11 at line 17, pg. 4 at line 18, pg. 4 at line 20, pg. 11 at line 17, pg. 5 at line 3, among others). As a consequence, it is unnecessary to further limit these method claims by describing in the claims, for instance, the relative timing of yields or the processes by which a subset of the plurality of threads was designated.

With regard to claims 16 and 31, these apparatus claims clearly recite structural elements for implementing embodiments of the invention, e.g., "a computer" and "a program." Moreover, the claims and rest of the specification sufficiently describe cooperative relationships between elements, such as "resources" on a "multithreaded CPU," and "threads" within a "common virtual space." The Examiner is urged to read at

least the section of the specification, entitled "Multithreaded CPU Yield," to find support for the above claimed processes and associated structures. Applicants therefore respectfully request withdrawal of all of the §112 rejections.

As a final matter in regard to several of the PTO/SB/08A's filed by Applicants, Applicants note that the initialed copy of the December 7, 2004 filing was not included with this Office Action, as indicated in the Attachments section of the Office Action Summary, nor does it appear in the image file wrapper file history on the PAIR system. A copy of this PTO/SB/08A is attached for the Examiner's convenience. If the Examiner needs a copy of the reference, the Examiner is strongly urged to contact the undersigned, and a copy will be immediately supplied to the Examiner. In addition, Applicants also noted that reference BR on the second page of the PTO/SB/08A filed on October 15, 2001 and returned with the October 19, 2004 Office Action was not initialed by the Examiner. Applicants have also included a copy of this paper. Again, if the Examiner needs a copy of the reference, he is urged to contact the undersigned. Furthermore, Applicants also noted that the Information Disclosure Statement mailed by Applicants on July 8, 2002 is also not included in the image file wrapper file history on the PAIR system. Applicants have attached a copy of this submission, including copies of the references, along with a copy of the return postcard indicating the Office's receipt of same on July 15, 2002. Applicants respectfully request that the July 15, 2002 filing be included in the PAIR system image file wrapper file history and that the Examiner return initialed copies of all of the above-included PTO/SB/08A forms to Applicants in the Examiner's next communication.

In summary, Applicants respectfully submit that all pending claims are novel and non-obvious over the prior art of record. Reconsideration and allowance of all pending claims are therefore respectfully requested. If the Examiner has any questions regarding the foregoing, or which might otherwise further this case onto allowance, the Examiner may contact the undersigned at (513) 241-2324. Moreover, if any other charges or credits are necessary to complete this communication, please apply them to Deposit Account 23-3000.

Respectfully submitted,

9/23/05
Date

Douglas A. Scholer
Reg. No. 52,197
WOOD, HERRON & EVANS, L.L.P.
2700 Carew Tower
441 Vine Street
Cincinnati, Ohio 45202
Telephone: (513) 241-2324
Facsimile: (513) 241-6234